Remarks:

Claims 1, 9, 13, 16, 20, and 23 are amended. Claims 1, 3-5 and 7-28 remain in the application.

ARGUMENT

Claims 1, 3, 4, 9, 10, 11, 13, 16, 17, 18, 20 and 23-25 are rejected under 35 U.S.C. § 103(a) as being unpatentable over USPN 6,397,242 to Devine (hereinafter "Devine") in view of "*The Technology Behind Crusoe Processors*" by Alexander Klaiber (hereinafter "Klaiber) and further in view of view of "*Complete Computer System Simulation: The SimOS Approach*", Winter 1995, IEEE Parallel & Distributed Technology, pages 34-43, by Rosenblum et al. (hereinafter, "Rosenblum"). This rejection is respectfully traversed and Claims 1, 3, 4, 9, 10, 11, 13, 16, 17, 18, 20 and 23-25 are believed allowable as amended based on the following discussion.

The Examiner asserts that Klaiber teaches *the monitor modifying the original values in a descriptor table to prevent the translated code from being accessed*... However, Klaiber teaches setting a bit in the page's entry in the processor's memory management unit (MMU). Klaiber then describes that this bit is invisible to X86 software. Klaiber's method modifies the actual page table and <u>not</u> a descriptor table.

In contrast, descriptor tables are known in the art, as are segment tables/descriptors. For instance, a definition for a segment descriptor may be found at en*wikipedia*org/wiki/Segment_descriptor that reads:

"In memory addressing for Intel x86 computer architectures, segment descriptors are a part of the segmentation unit, used for translating a logical address to linear address. Segment descriptors describe the memory segment referred in the logical address.

A logical address in Intel x86 consists of a segment selector and an offset. The most significant 13 bits of the segment selector defines the address of the segment descriptor, which is stored either in the Global Descriptor Table (GDT) or Local Descriptor Table (LDT). The description details mainly include the memory segment's first byte in linear address (Base), size (Limit) and Type (Bovet & Cesati, 2000, p. 36 - 41)."

Descriptor tables are also known in the art and described in the specification. The descriptor table with the segment table is described as being modified by the virtual machine monitor. The segment table in the descriptor table is not the same thing as a "page's entry in the MMU." Segmentation is one form of address translation that happens prior to paging. Page table entries are a separate mapping function.

Moreover, Klaiber teaches write-protecting a page of memory. The modification as claimed by Applicants prevents the translated code from being accessed, not an entire page of memory. As described by Applicant, segmentation provides a mechanism for dividing the processor's linear address space into smaller protected regions called "segments." The operating system defines its segments by assigning a segment base, a segment limit, and different segment attributes, e.g., type, granularity, DPL (Descriptor Privilege Level). In contrast, pages are defined by the MMU. Thus, the prior art fails to show each limitation of the recited claims.

Further, Rosenblum seems to teach an operating system simulator. In contrast, the claimed simulator supports a full platform simulator which, as described in the specification, includes simulation of the instruction set architecture of the processor.

Independent claims 1, 9, 16 and 23 have been amended to more clearly recite that segment information is modified in the descriptor tables. Thus, Claims 1, 9, 16, 23 and their progeny are believed allowable.

Claims 5, 12, 19 and 26 are rejected under 35 U.S.C. § 103(a) as being unpatentable over Devine, Klaiber, Rosenblum and further in view of "*Running multiple operating systems concurrently on an IA32 PC using virtualization techniques*" by Kevin Lawton (hereinafter "Lawton"). This rejection is respectfully traversed and Claims 5, 12, 19 and 26 are believed allowable as amended based on the foregoing and following discussion.

Claims 5, 12, 19 and 26 are believed allowable, at least by being dependent on allowable base claims.

Claims 7, 8, 14, 15, 21-22 and 27-28 are rejected under 35 U.S.C. § 103(a) as being unpatentable over Devine, Klaiber, Rosenblum and further in view of "*Operating systems* *internals and design principles*" by William Stallings (hereinafter "Stallings"). This rejection is respectfully traversed and Claims 7, 8, 14, 15, 21-22 and 27-28 are believed allowable as amended based on the foregoing and following discussion.

Claims 7, 8, 14, 15, 21-22 and 27-28 are believed allowable, at least by being dependent on allowable base claims.

Further, the Examiner asserts that Stallings teaches modifying the descriptor table to remove a portion of a segment that overlaps with the memory storing the translated code, or replace a segment to create a fault. However, it seems that Stallings merely teaches basic techniques for creating memory segments.

A cursory review of the teachings of Stallings do not show a teaching or suggestion that overlapping portions are to cause the descriptor table to be modified. It seems that Stallings is merely teaching how to set up a segmentation scheme and the translation of segment addresses. Stallings seems to teach allocating and translating segments and the handling of growing or shrinking data structures. There seems to be no teaching or suggestion for modifying the descriptor tables when segments overlap translated code. In fact, no mention of translated code seems to appear at all. Thus, Stallings fails to teach or suggest the claimed limitation.

The Examiner asserts that it would be obvious to modify Devine, Klaiber and Rosenblum with Stallings to modify a segment or replace a segment with a substitute segment to cause a fault. However, Stallings teaches only that more processes may be put in memory by overlaying them. This overlay scheme is based on original code, and not translated code. There is no suggestion in Stallings that translated code, i.e., code to be simulated, is to be written into data segments that cause an overlap of segments to the translated code. In contrast, the claimed invention translates machine instructions into translated code, and this translation may cause an overlap. The overlap here is not the same as overloading memory with too many processes in memory. The problem is different, the cause is different, and the result is different. Stallings will swap memory back and forth when a fault occurs.

In contrast, Applicants' invention transfers control to between a VM and VMM so that the translated code may be simulated. Thus, there is no suggestion in Stallings to replace segments for this purpose, nor would it be obvious to one of skill in the art. Moreover, as recited in the base claims, a modification of the segment information in the descriptor table is made to prevent the translated code from being accessed. This type of modification is neither taught nor suggested by Stallings. Thus, all of the pending claims are believed allowable.

CONCLUSION

In view of the foregoing, Claims 1, 3-5 and 7-28 are all in condition for allowance. If the Examiner has any questions, the Examiner is invited to contact the undersigned at (703) 633-6845. Early issuance of Notice of Allowance is respectfully requested. Please charge any shortage of fees in connection with the filing of this paper, including extension of time fees, to Deposit Account 50-0221 and please credit any excess fees to such account.

Respectfully submitted,

Dated: 13 Dec. 2007

/ Joni D. Stutman-Horn /
Joni D. Stutman-Horn, Reg. No. 42,173
Patent Attorney
Intel Corporation
(703) 633-6845

Intel Corporation
c/o Intellevate, LLC
P.O. Box 52050
Minneapolis, MN 55402